DE GRUYTER OPEN

Acta Geophysica
vol. 63, no. 5, Oct. 2015, pp. 1447-1462
DOI: 10.1515/acgeo-2015-0040

Remote Sensing Data Binary Classification Using Boosting with Simple Classifiers

Artur NOWAKOWSKI

Space Research Centre, Polish Academy of Sciences, Warszawa, Poland
e-mail: anowakowski@cbk.waw.pl

A b s t r a c t

Boosting is a classification method which has been proven useful in non-satellite image processing while it is still new to satellite remote sensing. It is a meta-algorithm, which builds a strong classifier from many weak ones in iterative way. We adapt the AdaBoost.M1 boosting algorithm in a new land cover classification scenario based on utilization of very simple threshold classifiers employing spectral and contextual information. Thresholds for the classifiers are automatically calculated adaptively to data statistics.

The proposed method is employed for the exemplary problem of artificial area identification. Classification of IKONOS multispectral data results in short computational time and overall accuracy of 94.4% comparing to 94.0% obtained by using AdaBoost.M1 with trees and 93.8% achieved using Random Forest. The influence of a manipulation of the final threshold of the strong classifier on classification results is reported.

Key words: land cover classification, boosting, artificial area, IKONOS.

1. INTRODUCTION

Contemporary commonly used methods of satellite image supervised classification are usually based on neural networks, decision trees or support vector machine (SVM) (Brito and Quintanilha 2012). The need for high accuracy of results leads to search for new approaches. One of them is boosting (Quinlan 1996), which has been proven useful in other applications, especially in biometrics. It is a technique which builds a strong classifier from many weak ones and is an example of ensemble classification.

Boosting was employed in a wide range of applications in computer vision for non-satellite imaging. For example, it was successfully applied in face detection (Viola and Jones 2004), license plate localization (Dlagnekov 2004), and biomedical datasets classification (Cerquides *et al.* 2006).

The technique has been already partially exploited in satellite image classification for the pixel-oriented approach. As for weak classifiers can be chosen any of contemporary classification methods (*e.g.*, thresholding spectral values, nearest-neighbourhood, neural network, decision trees, support vector machine, Bayesian methods, *etc.*), the vast majority of satellite image studies employ boosting with decision trees methods (Schneider *et al.* 2010, Chan and Paelinckx, 2008, Lawrence *et al.* 2004, Friedl *et al.* 1999, 2002; McIver and Friedl 2001). They are mostly based on C4.5/C5.0 algorithms and tools implemented in C/C++ (or J4.8 – their Java version) developed by Quinlan (1996). Only few studies are related to utilization of boosting with weak classifiers other than decision trees. Briem *et al.* (2002) and then Benediktsson *et al.* (2007) compared performance of different classification methods including simple one-feature algorithm 1R (Holte 1993), decision table and decision tree to their ensemble versions, where these methods were used as weak classifiers. They used three ensemble methods: boosting, bagging, and consensus theory. In the comparison, authors also included other popular classifiers like minimum Euclidean distance, Gaussian maximum likelihood, and neural network. Performance results on two multisource remote sensing and geographic datasets indicate that boosting classifiers outperforms other ones: for the first datasets the highest overall and average test accuracies were obtained for boosting with simple one-feature 1R classifiers while for the second dataset boosting with decision trees was the best.

In this paper we investigate a novel scenario for land cover classification for pixel-oriented approach, where as weak classifiers for boosting we use simple threshold classifiers based not only on spectral values of the pixel, but also on statistical and contextual information. The weak classifiers are defined adaptively to training data. The method can be applied for any land cover binary classification problem, but in order to prove its high performance we tested the method on non-trivial exemplary classification problem.

As the example we selected artificial area identification on IKONOS multispectral data.

2. BOOSTING IDEA

The idea of creating a strong classifier from many weak ones has been investigated since the mid-1980s (Valiant 1984, Kearns 1988). The first algorithm of boosting classification, which became popular because of its reliability, was developed by Y. Freund and R. Schapire in 1995 and called AdaBoost (Freund and Shapire 1996). Since then, some variants of that method have been introduced as well as many other boosting algorithms have been proposed (Matas and Sochman 2001).

Boosting is a supervised method and it assumes availability of a set of training samples. It has two phases of processing: training and testing. During training procedure, a strong classifier is built based on training data. Testing serves for classification of all image pixels using obtained strong classifier.

The common approach to the training stage of boosting methods is to build a strong classifier from iteratively selected weak classifiers. In each iteration, every weak classifier is evaluated on weighted training data and a classification error is provided. The weak classifier which produces the smallest error is added to the resulting strong classifier with computed weight. In the same iteration, weight of every training sample which has been wrongly classified by the selected weak classifier is increased, while the weights of other samples are decreased. This operation allows to concentrate the algorithm on problematic samples in next iteration. The response of the resulting strong classifier K is a weighted linear combination of responses of selected weak classifiers k_i, $i = 1, ..., N$:

$$K = \omega_1 k_1 + \omega_2 k_2 + \omega_3 k_3 \ , \tag{1}$$

where $\omega_l \in R$: (0,1] denotes weights. This strong classifier is directly used in the testing stage.

Adaptation of the boosting algorithm to a classification problem requires: (i) selection of a feature space, (ii) construction of a set of weak classifiers, (iii) definition of a classification error measure, and (iv) selection of training samples.

3. FEATURE SPACE

Spectral information is the most frequently selected feature for pixel-oriented classification in decision trees with boosting solutions (Schneider *et al.* 2010, Chan and Paelinckx 2008, Lawrence *et al.* 2004) and other approaches (Brito and Quintanilha 2012). Other features like textural, geomet-

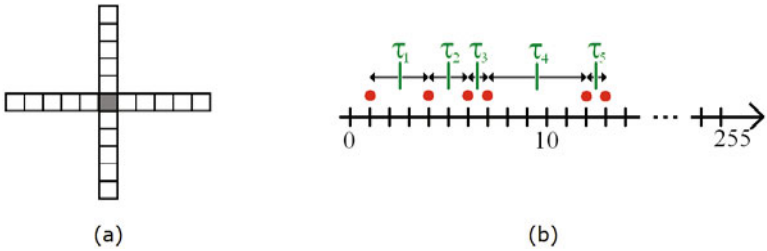


Fig. 1. Neighbourhood window for the actual pixel (localized in the middle) (a), and threshold selection in a binary histogram, where circle denotes that a certain value exists in training data and τ_i are necessary thresholds (b).

tric or topologic information are used rarely. They are applied mainly in object-oriented classification (Brito and Quintanilha 2012).

In our method devoted to land cover classification we propose to utilize all available spectral bands and create one additional band containing mean value of all image band values. Current trends show that besides spectral values of the pixel, spatial information could help in classification (Benediktsson *et al.* 2007). Therefore, we include a few basic statistics in the neighbouring window of each considered pixel for each spectral band:

- mean value,
- variance,
- maximal spectral difference (the difference between maximum and minimum).

The shape of the neighbouring window is designed to meet the opposite criteria: minimal size and simplicity (to limit computing time), and maximum information about the neighbourhood (to maximize accuracy). In order to fulfil these, we resigned from using classical square window and left only these neighbouring pixels from it which contained information in four main directions, obtaining the cross-shaped window (Fig. 1a). All the resulting statistic values are scaled linearly to the same discrete range as spectral values to unify the codomain and to reduce computation cost. The size of the window has been set to 11 pixels in horizontal and vertical directions. For dealing with the edge pixels we chose mirroring approach, however other methods could also be applied.

The total number of features amounts to $4 * (N_b + 1)$, where N_b denotes the number of bands in an imagery.

4. WEAK CLASSIFIERS

For each selected feature we consider a set of all possible threshold classifiers. For a single feature with N possible discrete values there exist $N - 1$ possible values for the threshold, and therefore $N - 1$ different threshold

classifiers can be defined. Considering all of the features leads to very large total number of weak classifiers. For example, for IKONOS 4-band 8-bit imagery we get 20 features and therefore obtain $20 \times 255 = 5100$ possible thresholds assuming that every non-spectral feature has a range of 8 bits.

Each threshold is a base to define two classifiers: one determining whether a particular class has values equal or less than the threshold, the second determining whether this class has values above it. Therefore, the number of weak classifiers is equal to double the number of thresholds. For considered example, we achieved over 10^4 weak classifiers.

However, it is probable that real data does not contain all of the theoretically possible values. Therefore, in order to reduce the total number of weak classifiers, only the thresholds necessary in the analysed case are considered (Fig. 1b). For every feature, the algorithm builds a binary histogram which has a positive value for those theoretical feature values which exist in data, and a zero value in opposite case. Subsequently, the thresholds are defined as mean values of two neighbouring non-zero values in the histogram.

A set of all found classifiers based on feature of one kind (among spectral values, mean values in window, variances in window or maximal difference in window) is hereinafter called a family of weak classifiers. Defined families of weak classifiers are used by boosting algorithm in order to build one strong classifier.

5. CLASSIFICATION METHOD

The method assumes availability of a training set of pixels classified into two classes. During the whole process, manual work is needed only for the creation of the training set. Training and testing phases are fully automatic.

5.1 Training

In the first step, for each training point, the features described in Section 3 are computed and the set of necessary thresholds for each feature is calculated according to Section 4. The list of determined thresholds is an input to the second step which applies the boosting algorithm. We chose AdaBoost.M1 algorithm due to its simplicity and reliability (Freund and Schapire 1996). Among few possible implementations we choose the classical one of Viola and Jones (2001). It assumes that positive training samples are classified as 1 while the negative training samples as 0. The algorithm starts with the initialization of sample weights according to their positive or negative type ($w_{p,i}$ or $w_{n,j}$):

$$w_{p,i} = \frac{1}{2N_p}, \quad i = 1,...,N_p \ , \tag{2}$$

$$w_{n,j} = \frac{1}{2N_n} \quad j = 1,...,N_n \ , \tag{3}$$

where N_p and N_n denote the number of positive and negative samples, respectively.

Next, the main boosting loop begins. In each iteration a number of steps are executed and sample weights are normalized by division by the sum of all weights (including weights of positive and negative samples). After this operation all the weights sum up to 1 and can be interpreted as probability distribution. Subsequently, all weak classifiers (defined using input thresholds) in all families are consecutively tested on weighted training data. For each feature and each of its thresholds τ_k^f, $k = 1,...,K^f$, $f = 1,...,F$ (K^f denotes the number of thresholds for feature f, F denotes number of features) two classification errors, $e_0(k,f)$ and $e_1(k,f)$, are computed. The first one is devoted for the weak classifier where the first class is considered to be below the threshold, the latter one being dedicated for the opposite case. They are calculated as a sum of weights of misclassified training samples w_i:

$$e_{k0}(k,f) = \sum_{i=1}^{n} w_i, \quad i = 1,...,N_k^f \ , \tag{4}$$

$$e_{k1}(k,f) = 1 - e_{k0} \ , \tag{5}$$

where N_k^f denotes the number of misclassified pixels using the weak classifier of feature f and threshold k and the first class is considered to have values below the threshold.

As we need to select the best weak classifier, minimal error $e_{\min}$ is found in a set of values e_{k0} and e_{k1} for all thresholds for all features:

$$e_{\min} = \min_{k=1,...,K^f, f=1,...,F} \left\{ e_{k0}(k,f), e_{k1}(k,f) \right\} \ . \tag{6}$$

During consecutive evaluations we preserve only the value of the smallest error already computed, related feature number, threshold value and binary information whether this is e_{k0} or e_{k1}.

Next, we update the weights of training samples which were correctly classified by the corresponding classifier:

$$w_i' = \frac{e_{\min}}{1 - e_{\min}} w_i \ . \tag{7}$$

We do not modify the weights of wrongly classified samples directly, but in fact they are increased in the next loop because of the normalization step. In the last step of the loop, the found best weak classifier k_j is added to the resultant strong classifier with the calculated weight ω_j:

TRAINING

Inputs:

1) I_L - an image with L channels,
2) $p_i \in \mathbf{N}^2$, $i=1,..,N_p$, $n_i \in \mathbf{N}^2$, $i=1,..,N_n$ - localisations of positive (valued as 1) and negative (valued as 0) samples
3) N_s - maximum number of steps,
4) e_m - maximum classification error

Method

I. Compute features:

1) For each pixel $i \in I_L$ for each spectral value and the mean of all of them compute the following additive features: mean value, variance, maximal spectral difference.
2) For every feature f normalise values for all training samples into range [0,255].

II. Select classification thresholds

For each feature f:

1) compute binary histogram of values,
2) select thresholds t_k^f, $k=1,...,K^f$

III. Classify data using AdaBoost.M1:

1) Initialize weights of samples:
$w_{p,i} = \frac{1}{2N_p}, i = 1,\dots,N_p, \qquad w_{n,j} = \frac{1}{2N_n}, j = 1,\dots,N_n$
2) For $s=1,...,N_s$ do:
   a) normalize weights of samples: $w_{p,i} = \frac{w_{p,i}}{\sum_{j=1}^{N_p} w_{p,j} + \sum_{j=1}^{N_n} w_{n,j}}$,
   $w_{n,i} = \frac{w_{n,i}}{\sum_{j=1}^{N_p} w_{p,j} + \sum_{j=1}^{N_n} w_{n,j}}$,
   b) for each feature f and each of its thresholds t_k^f, compute two classification errors $e_0(k,f)$ and $e_1(k,f)$ on weighted samples,
   c) select the classifier k_s resulting in the smallest value e_{min} and add it to the strong classifier with the weight:
   $\omega_j = log\frac{1-e_{min}}{e_{min}}$
   d) if $e_{min} < e_m$ then break,
   e) update the weights of correctly classified samples by k_s:
   $w'_i = \frac{e_{min}}{1-e_{min}} w_i$, 3

IV. Normalize the strong classifier:

1) For each weak classifier normalise its weight:
$\omega_j = \frac{\omega_j}{\sum_{j=1}^{J} \omega_j}$

V. Shorten the strong classifier

Output: Normalised strong classifier: $K=\omega_1 k_1 + \omega_2 k_2 + \dots + \omega_J k_J$

TESTING

Inputs:

1) I_L - an image with L channels,
2) K - normalised final strong classifier

Method

For every pixel $i \in I_L$:

1) For each spectral value and the mean of all of them compute the following additive features: mean value, variance, maximal spectral difference.
2) Classify pixel using K:
   if $K(i) = \sum_{j=1}^{J} \omega_j k_j(i) \geq 0.5$ then mark pixel as positive (1) else mark as negative (0)

Output: Classified image I_L

Fig. 2. Pseudocode of the proposed classification method.

$$\omega_j = \log \frac{1 - e_{\min}}{e_{\min}} \ . \tag{8}$$

The loop of the algorithm is continued and in each iteration the best weak classifier is selected. The process stops if the classification error $e_{\min}$ is less than a set value or the number of iteration exceeds assumed level. The resultant strong classifier is a weighted sum of all the selected weak classifiers according to Eq. 1. The last step is a normalization of weak classifiers' weights:

$$\omega_j = \frac{\omega_j}{\sum_{j=1}^{J} \omega_j} \ . \tag{9}$$

The rate of the convergence in iterations of the method to minimal classification error strongly depends on data and employed types of weak classifiers and is difficult to predict. In general, the rate should increase with dimensionality of the feature space and with the use of more accurate weak classifiers. On the other hand, our experiments have shown that in some cases, after the algorithm converged to the minimum classification error, the error could start to slowly increase in next iterations. Therefore, if the method stopped because the number of iteration exceeded the assumed level, the strong classifier is shortened according to theloop for which the minimal error was achieved.

The training phase of the method is summarized in pseudocode in Fig. 2.

5.2 Testing

Assuming that a weak classifier denotes pixel belonging to the first class as 1, and the rest of pixels as 0, pixel's class is based on determination if the response of the strong classifier is below or above the half of the sum of all weights of the weak classifier. The sum of all weights is equal to 1 because of normalization step (Eq. 9). According to Eq. 1, a pixel p can be classified as belonging to the first class if the following condition is true:

$$K(p) = \sum_{i=1}^{J} \omega_j k_j(p) > 0.5 \ . \tag{10}$$

Elsewhere, the pixel p is considered to be a member of the second class. The testing phase of the method is summarized in pseudocode in Fig. 2.

6. IMPLEMENTATION

Implementations of basic boosting algorithms could be found in some statistical tools or programming libraries. However, in order to have an operational implementation allowing full insight into the process with a possibility to modify it and to access partial results, we implemented the method in

C++. All the processing steps of the method are done automatically for both training and testing phases. We employed GDAL library (http://www.gdal.org) for handling the input and output. It allows us to work with many different image file formats.

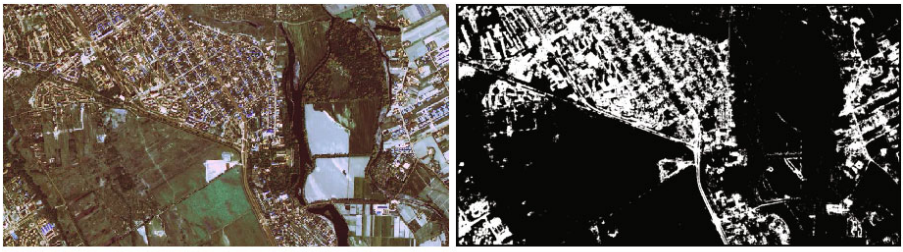
7. EXPERIMENTS

7.1 Classification problem

In order to show that the introduced method can be successfully used for satellite data classification it is tested for a problem of artificial area identification on multispectral IKONOS imagery. For such data, precise classification of artificial areas is a challenge due to spectral diversity of the class. Target areas include roads, pavements, buildings, car parks, *etc.* They could be in size of few or even sub-pixel. In order to achieve high level of information about details and to not omit any target areas we focus on one pixel accuracy of classification. In the case of larger minimal mapping unit, it is possible to use existing methods of aggregation in postprocessing of the results obtained with the method described here.

7.2 Test data

We defined a fragment of 8-bit multispectral IKONOS imagery with spatial resolution of 4 m, sized 1100×600 pixels, as an experimental area which depicts southern part of the City of Warsaw (Fig. 3a). It includes a variety of types of artificial areas: sparse and dense, high-rises and low-rises with big and small roof areas, regular and irregular shapes and different roof cover, different kinds of roads, car parks, pavements, construction sites, cemeteries, *etc.* On the other hand, the test image contains examples of various types of other classes: water, forests, bushes, grasslands, meadows, green parks, bare soil, *etc.* By visual inspection we prepared 1643 training pixels selected as single ones or in polygons of which 767 pixels belong to the artificial area class (positive samples) and 876 pixels to the non-artificial area class (negative samples).



(a) (b)

Fig. 3. Artificial area identification: (a) original IKONOS data, and (b) classification results.

7.3 Classification results

For the training phase an important issue is the speed of convergency of the method to the minimum classification error. Figure 4 shows how the classification error decreases in consecutive iterations of a boosting loop for the training data. As we defined that this error cannot exceed 0.003, the boosting method stopped because of the second stop criterion – the maximal number of iterations (set to 200). This number should not be too high in comparison to number of training pixels in order to avoid overfitting. The whole training process on middle class PC (Intel Core2Quad CPU Q9300, 2.5GHz, 4MB) took 6 s.

Results of the testing phase (classification of the whole test area) are presented in Fig. 3b (computational time was about 1 min). The method shortened a strong classifier to 197 weak classifiers, for which the classification error on the training data achieved the smallest value of 0.0067.

Generally, the achieved results should be marked as very good considering that the image was classified into only two heterogenic classes. Visual analysis highlighted two main misclassification problems. Firstly, borders of some wetland were detected as artificial areas. Secondly, few bare soil areas were detected also as artificial, which is probably connected to the fact that some examples of construction sites included in artificial areas training set were during ground works.

In order to obtain authoritative assessment of the classification results, the validation procedure was applied. We used a set of 2000 validation pixels, different from the training ones, selected by visual inspection including analysis of data from other imageries. Their localization was chosen randomly using standard uniform distribution over the whole test data. As we aimed at pixel level accuracy, we were confronted with the problem of classification of pixels which are not unimodal (contain both classes) or pixel repre-

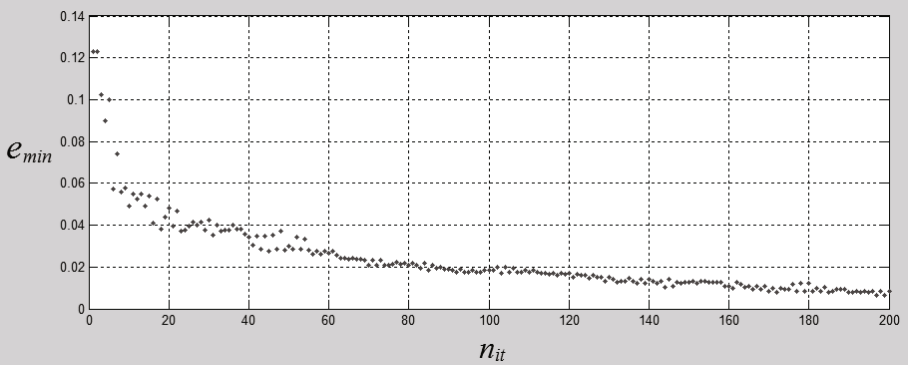


Fig. 4. Classification error ($e_{\min}$) *versus* number of boosting loop iterations (n_{it}).

senting undetermined land cover. Therefore, we defined 3 classes that a pixel could represent: 1 – pure artificial area, 2 – non-classified or mixed, 3 – pure non-artificial area. Statistics of the validation set are presented in Table 1.

Table 1

Validation classes and statistics

Class no.	Class description	Number of pixels
1	Pure artificial	283
2	Non-classified or mixed	424
3	Pure non-artificial	1293
	Total:	2000

The overall accuracy achieved 94.4%. Such high accuracy is due to spatial dominancy of non-artificial areas in the image. Nevertheless, producer and user accuracies for artificial area class are still high (see details in Table 2).

Table 2

Validation results

Accuracy	Pure artificial [%]	Pure non-artificial [%]
User	82.3	97.1
Producer	86.0	96.2
Overall	94.4	

Using the same training and validation data and the same features we computed classification accuracies for two other ensemble methods: AdaBoost.M1 with decision trees as weak classifiers and Breiman's Random Forest' algorithm (we used implementation of the methods from Statistics Toolbox in Matlab R2012a). For both methods we prepared classification for different numbers of weak classifiers. The obtained overall accuracies in comparison to results of the proposed method are presented in Fig. 5. Computations for the proposed method stopped for the number of 354 weak classifiers when the training error achieved 0 value. Therefore setting up more steps resulted in the same accuracy, which was the best one. Other two methods achieved slightly worse maximums of accuracies equal to 94.0% (AdaBoost with trees) and 93.8% (Random Forest). On the other hand, differences in computational efforts are very significant. Considering only the

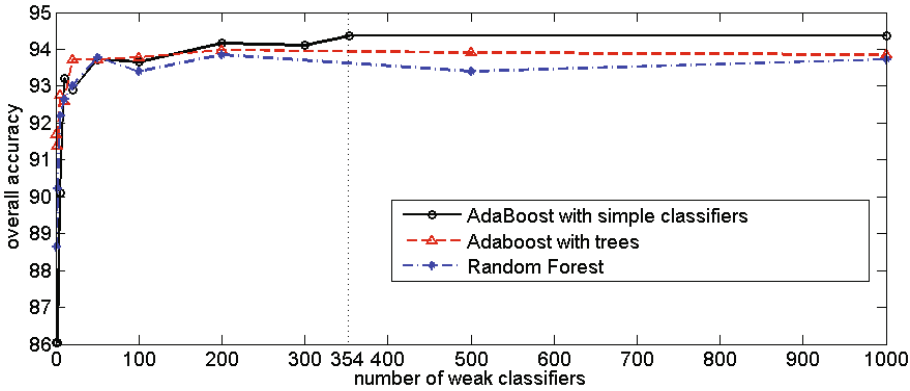


Fig. 5. Overall accuracy on validation set *versus* number of weak classifiers.

testing phase of the classification, when the minimisation of the efforts is critical due to the necessity of processing all pixels in an imagery, the proposed method needs only one threshold comparison for each weak classifier. AdaBoost.M1 employing trees and Random Forest need as much threshold comparisons as the number of internal nodes is in each decision tree. For the results presented in Fig. 5, the AdaBoost employing trees produced trees with mean number of internal nodes equal to 57, while for Random Forest this number is 63. Thus, the difference in needed comparisons is several dozen. Moreover, storing, managing and even interpretation of strong classifier is much easier in the proposed method, because the structure of proposed strong classifier is less complex.

7.4 Final threshold manipulations

In the boosting method the final classification decision is done using a threshold 0.5 (Eq. 10). In theory, this coefficient is utilized in order to minimize the error (*e.g.*, misclassification rate). However, the coefficient can also be seen as a parameter which can be changed in order to manipulate classification accuracies. According to Eq. 10:

$$K(p) = \sum_{i=1}^{J} \omega_j k_j(p) > T, \quad T \in (0,1) \ . \tag{11}$$

In Figure 6 we present the results of such a manipulation for the considered problem after validation. Although we expected to achieve the highest overall accuracy for $T_1 = 0.50$, it was obtained for $T_2 \approx 0.51$ (Fig. 6a). This difference, almost negligible, improves the overall accuracy by only about 0.3% to 94.7%.

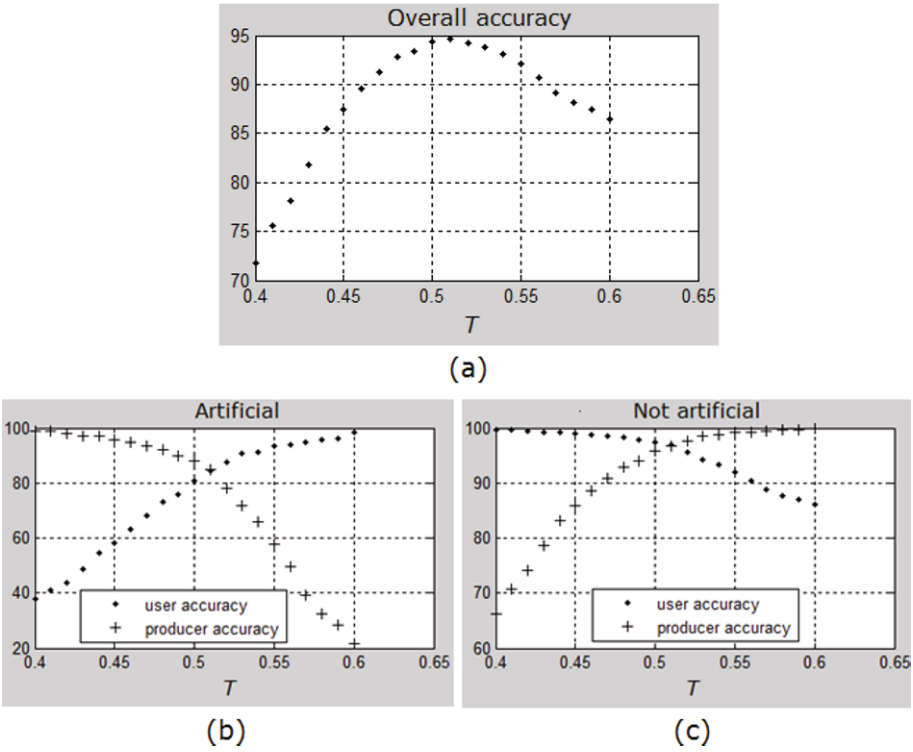


Fig. 6. Classification accuracy *versus* final threshold of the method: overall accuracy (a), user and producer accuracies for artificial (b) and non-artificial (c) surfaces.

Nevertheless, the manipulation of T can be useful to change the compromise between user and producer accuracies for a certain class (Fig. 6b-c). Depending on an application of the classification it is sometimes desired to prefer one kind of these accuracies over another. For the classification error computation employed here (sum of weights of misclassified training samples) it is clear that user and producer accuracy for both artificial and non-artificial surfaces are equal when the maximum of overall accuracy is achieved at T_2. At that time, the accuracy for artificial surfaces amounted to 84.5% while for non-artificial ones it was 96.8%.

8. DISCUSSION AND CONCLUSIONS

We introduced a novel satellite image classification method which is based on the boosting classification. Simple threshold classifiers are used as weak ones instead of popular decision trees. We chose all possible threshold classifiers based on spectral values of pixels and basic statistics of pixel cross-

shaped neighbourhood. The numbers of threshold classifiers are minimized based on data histograms.

The method is a supervised technique and requires a training set. However, training and testing are fully automatic, once the two parameters for stop criteria are defined. It could also be used in a semi-automatic way for the case when user is not interested in maximizing the overall, but only user or producer accuracy. In that case, the final threshold of the method could be manipulated in order to meet user's needs.

The method was implemented as stand-alone independent software and obtained computational time was short for both training and testing.

The performance was tested on a difficult problem of artificial area identification on IKONOS multispectral imagery. The obtained classification accuracy is 94.4%, which is slightly better than accuracies obtained by using other two state-of-the-art ensemble methods: AdaBoost.M1 with decision trees and Random Forest. Achieved results together with evaluations introduced with Briem *et al.* (2002) and then Benediktsson *et al.* (2007) prove the thesis that there is no need to use advanced weak classifiers to obtain high accuracy of classification results.

As the boosting scenario which is based on advanced weak classifiers has already been partially exploited in satellite image analysis, the scenario with very simple classifiers is a novel one and need further investigations, in particular in the area of performance with other types of features and classifiers, and in adaptation to other practical classification problems.

Acknowledgments. This work was supported by ESA under the framework of PECS, project No. 4000105537/12/NL/KML: Implementation of remote sensing data and models in optimizing the localisation of renewable energy sources on the example of biofuel crops with respect to ecological constraints.

References

Benediktsson, J.A., J. Chanussot, and M. Fauvel (2007), Multiple classifier systems in remote sensing: from basics to recent developments. **In:** M. Haindl, J. Kittler, and F. Roli (eds.), *Multiple Classifier Systems*, Lecture Notes in Computer Science, Vol. 4472, Springer, Berlin Heidelberg, 501-512, DOI: 10.1007/978-3-540-72523-7_50.

Briem, G.J., J.A. Benediktsson, and J.R. Sveinsson (2002), Multiple classifiers applied to multisource remote sensing data, *IEEE Trans. Geosci. Remote Sens.* **40**, 10, 2291-2299, DOI: 10.1109/TGRS.2002.802476.

Brito, P.L., and J.A. Quintanilha (2012), A literature review, 2001-2008, of classification methods and inner urban characteristics identified in multispectral remote sensing images. **In:** *Proc. 4th GEOBIA, 7-9 May 2012, Rio de Janeiro, Brazil*, 586-591.

Cerquides, J., M. López-Sánchez, S. Ontañón, E. Puertas, A. Puig, O. Pujol, and D. Tost (2006), Classification algorithms for biomedical volume datasets. **In:** R. Marin, E. Onaindia, A. Bugarin, and J. Santos (eds.), *Current Topics in Artificial Intelligence*, Springer, Berlin Heidelberg, 143-152, DOI: 10.1007/11881216_16.

Chan, J.C.W., and D. Paelinckx (2008), Evaluation of Random Forest and AdaBoost tree-based ensemble classification and spectral band selection for ecotope mapping using airborne hyperspectral imagery, *Remote Sens. Environ.* **112**, 6, 2999-3011, DOI: 10.1016/j.rse.2008.02.011.

Dlagnekov, L. (2004), License Plate Detection using AdaBoost, Computer Science and Engineering Department, University of California, San Diego, USA.

Freund, Y., and R.E. Schapire (1996), Experiments with a new boosting algorithm. **In:** *Proc. 13th Int. Conf. Machine Learning, 3-6 July 1996, Bari, Italy*, 148-156.

Friedl, M.A., C.E. Brodley, and A.H. Strahler (1999), Maximizing land cover classification accuracies produced by decision trees at continental to global scales, *IEEE Trans. Geosci. Remote Sens.* **37**, 2, 969-977, DOI: 10.1109/36.752215.

Friedl, M.A., D.K. McIver, J.C.F. Hodges, X.Y. Zhang, D. Muchoney, A.H. Strahler, C.E. Woodcock, S. Gopal, A. Schneider, A. Cooper, A. Baccini, F. Gao, and C. Schaaf (2002), Global land cover mapping from MODIS: algorithms and early results, *Remote Sens. Environ.* **83**, 1-2, 287-302, DOI: 10.1016/S0034-4257(02)00078-0.

Holte, R.C. (1993), Very simple classification rules perform well on most commonly used datasets, *Mach. Learn.* **11**, 1, 63-91, DOI: 10.1023/A:1022631118932.

Kearns, M. (1988), Thoughts on hypothesis boosting, University of Pennsylvania, Machine Learning class project, 105 pp. (unpublished).

Lawrence, R., A. Bunn, S. Powell, and M. Zambon (2004), Classification of remotely sensed imagery using stochastic gradient boosting as a refinement of classification tree analysis, *Remote Sens. Environ.* **90**, 3, 331-336, DOI: 10.1016/j.rse.2004.01.007.

Matas, J., and J. Sochman (2001), Adaboost. Center for Machine Perception, Czech Technical University, Prague, Czech Republic.

McIver, D.K., and M.A. Friedl (2001), Estimating pixel-scale land cover classification confidence using nonparametric machine learning methods, *IEEE Trans. Geosci. Remote Sens.* **39**, 9, 1959-1968, DOI: 10.1109/36.951086.

Quinlan, J.R. (1996), Bagging, boosting, and C4.5. **In:** *Proc. 13th National Conference on Artificial Intelligence, 12-17 February 1996, Phoenix, USA*, Vol. 1, 725-730.

Schneider, A., M.A. Friedl, and D. Potere (2010), Mapping global urban areas using MODIS 500-m data: New methods and datasets based on 'urban ecoregions', *Remote Sens. Environ.* **114**, 8, 1733-1746, DOI: 10.1016/j.rse.2010.03.003.

Valiant, L.G. (1984), A theory of the learnable, *Commun. ACM* **27**, 11, 1134-1142, DOI: 10.1145/1968.1972.

Viola, P., and M. Jones (2001), Rapid object detection using a boosted cascade of simple features. **In:** *Computer Vision and Pattern Recognition CVPR 2001*, Vol. 1, I-511–I-518, DOI: 10.1109/CVPR.2001.990517.

Viola, P., and M.J. Jones (2004), Robust real-time face detection, *Int. J. Comput. Vision* **57**, 2, 137-154, DOI: 10.1023/B:VISI.0000013087.49260.fb.

Received 26 June 2014
Received in revised form 27 November 2014
Accepted 5 January 2015